LOGIQ

How to Configure Kubernetes RBAC, Network Policies, and Runtime Privileges

INTRODUCTION

Kubernetes automates the application life cycle! But, how do you configure roles, capabilities, and control access rights in Kubernetes?

The goal of this guide is to show how you can secure a Kubernetes cluster using Roles Based Access Control (RBAC), Network Policies, and Runtime Privileges for maximum control and data privacy.

CONFIGURING K8S NETWORK POLICIES

Kubernetes is an open-source cluster management system that has a lot of applications in the world of business and commerce. Kubernetes was first developed by Google in 2014 and was heavily influenced by Google's Borg cluster management system. Currently, it is developed and maintained by the Cloud Native Computing Foundation.

The main benefit that Kubernetes lends to businesses is enhanced scalability. Depending on the load that your business is facing, you can scale the infrastructure up and down. This kind of scalability is extremely helpful and it can also save you a fortune. Kubernetes is the future of cluster management.

78% of all companies running containers have fully or partly deployed Kubernetes in their own infrastructure.

According to a 2020 report by the CNCF, 78% of all companies running containers have fully or partly deployed Kubernetes in their own infrastructure. By going through the tips listed in this article, you will be able to start configuring k8s network policies and pod security policies on your own.

LOGIQ

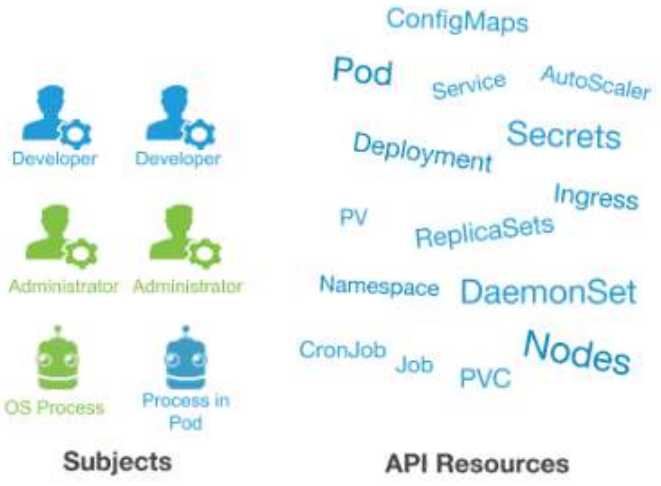
HOW TO CONFIGURE RBAC IN YOUR KUBERNETES CLUSTER

In this section of the guide, you'll learn how to configure RBAC in your Kubernetes cluster. Before getting all technical, you should be aware of two elements - Roles and ClusterRoles. The former describes rules that are applicable in a single namespace while the latter describes rules that are applied to the cluster.

USE CASE - CREATING A CLUSTER USER WITH LIMITED NAMESPACE ACCESS

The main use case of role-based access control within the Kubernetes environment is one where you create a user with limited namespace access. You have to memorize the following steps.

Step 1: You have to create the office namespace. To do so, you have to execute the 'kubectl create' command in the CLI to create a namespace for the office. To make sure this happens without any glitches, you have to do this while you are in cluster admin mode.



Step 2: In this step, you have to create the user credentials and grant them access based on their roles. Since Kubernetes network policy does not permit any API objects for user accounts, you will need to install OpenSSL certificates. You need to first create a key for your user.

Step 3: With the private key you created for the user, you have to come up with a certificate sign request on the CLI. The command has a 'subj' section where you have to specify the group and username. Next, you need to locate your Kubernetes network certificate authority (CA).

Step 4: Once you receive the final certificate, you will have successfully assigned a cluster user with proper role-based access. The certificate that you issued is going to have a validity of 500 days. Next, you need to create a role, bind it to the user, and test the RBAC.

Step 5: The final step in this process is where you have to create a role deployment file by using the 'kubectl create role' command. Then you can bind the specific role to the specific user by using the 'kubectl create' to bind the role file to the employee file.

LOGIQ

HOW TO CONFIGURE KUBERNETES NETWORK POLICIES

In this section, you'll learn how to go about configuring k8s network policies and pod security policy. The Kubernetes NetworkPolicy API is the best way for you to handle the task. You can apply these policies using cluster playgrounds like Minikube, Katacoda, and more.

Step 1: At first, you have to create an nginx deployment. Nginx is a web server that is mainly used as a reverse proxy. Use the 'kubectl create' command to declare an nginx deployment. Then, you need to expose the service that you have created by using an nginx service.

Step 2: Next, you have to test and see if the service is working by testing it from another pod. If the service is working, then it will be accessible via other pods. Start a busybox container from the pod and then run this command: 'wget -- spider -- timeout=1 nginx".

Step 3: Once the service is up and running, you have to limit access to the service. You have to assign a condition so that only pods with an 'access:true' label can access it. To do this, you have to create a NetworkPolicy object with a valid DNS name.

Step 4: The last step of the configuration process is where you assign the policy to the service. Using the 'kubectl create' command, the policy file is bound to the nginx service. Once this is done, it is also a good idea to check whether the access control is functional.

Make sure you've configured a network provider with network policy support. There are a number of network providers that support NetworkPolicy

HOW TO CONFIGURE RUNTIME PRIVILEGES

This final section of this guide will show you the best way to configure container runtime. In the Kubernetes framework, a container always runs inside a pod. The runtime policy of each pod can be achieved using Pod Specification, PSP, Open Policy Agent Gatekeeper, etc.

The best way to take care of container runtime privileges is by configuring a security context for the pod and the container it carries. You can implement Discretionary Access Control, App Armor, Seccomp, Linux Capabilities, Security-Enhanced Linux, and more. Setting the security context will ensure pod security policies also.

Once you are done with the configuration process, you need to clean up after yourself. The best way to do this is to use the 'kubectl delete' command to clear up all of the demo files created during the runtime privileges assignment. Once this is over, your work is complete!

Final Thoughts

It is important to remember that the word "kubernetes" actually comes from the Greek word for "helmsman". This is essentially what it does. It acts as a helmsman and a navigator for your cluster and keeps the system running smoothly.

With the help of this guide, you can now configure RBAC in your Kubernetes cluster, configure Kubernetes network policy and ensure that runtime is smooth. Kubernetes is a powerful tool that will enable your organization to work efficiently and safely. Have it installed ASAP and reap all of the benefits.

LOGIQ